Real-Time Human Activity Recognition from Wireless Sensors using Evolving Fuzzy Systems

Javier Andreu, *Student Member, IEEE*, Plamen Angelov, *Senior Member, IEEE*

***Abstract*— A new approach to real-time knowledge extraction from streaming data generated by wearable wireless accelerometers based on self-learning evolving fuzzy rule-based classifier is proposed and evaluated in this paper. Based on experiments with real subjects we collected data from 18 different classifieds activities. After preprocessing and classifying data depending on the sequence of activities regarding time, we achieved up to 99.81% of accuracy in recognizing a sequence of activities. This technique allows re-training the system as long as the application is running on the wearable intelligent/smart sensor, getting a better classification rate throughout the time without an increase of the delay in performance.**

I. Introduction

THE use of ubiquitous systems which monitor individuals is projected to increase rapidly in the next decade. The computer of the past was in a form of a rack, later the 'personal computer' was hailed as a great step in the miniaturization and personalization indeed, then laptops, PDAs and palm held computers become the next step in the same direction. The future computers will be micro and even nano devices that are literally embedded in various objects of the everyday life, including sports cloths, equipment used by elderly etc. All this miniaturized devises will surround us and will be able to monitor our movements, temperature or distance in a passive manner. MEMS and nano-electronics marked a significant advance in the increase of the processing capacities of these devices during the last two decades [1].

In the case of Wireless Sensors which are more suitable for monitoring purposes than mobile phones, sensory capacities have increased including several sensors that gather data in a synchronized fashion by comprising a powerful and very informative sensor array [2]. As a result of severe power consumption constraints, these wireless sensors limited in the past to merely gather the data from transducers and to forward it to the back-end. Nowadays, they are able to process complex algorithms, to compress, process the gathered data on-line.

This offers the possibility to develop principally new types of applications that are relatively more computationally complex but lead to the increase of the level of intelligence machine quotient (MIQ) of these devices and the systems and networks that include them. One such possibility to make use of the increased computational capacity of these devices for the development of an *intelligent* application is presented in this paper. It should be noted that despite the increased computational capacities the practical limitations that still exists as well as the requirements for *real-time* mode of operation require the algorithms and applications that are being developed to be implemented and uploaded on board such devices to be computationally lean, recursive and fast.

We have chosen as a case study of this generic approach activity recognition and alert generation task that is applicable to sportsmen elderly, soldiers etc. Activity recognition based on motion data is a very attractive field of study and experimentation for sport and healthcare applications. Monitoring wireless wearable devices can be embedded on everyday accessories such as bracelets or wrist watches which would be able to track all data at the same time offering a supposed high level meaning of the perceived information to a back-end center. At a (usually remote) centre/back-end coaches or doctors would be able to know the activity the subjects they are monitoring are most likely to be performing (involved in) as well as to receive an alert in case of an abnormal behavior. This information is the most important and can increase the efficiency of the monitoring process, save bandwidth and energy of transmission. Instead of sending the raw data and overloading the communication channel, monitors and the power requirements of the micro- or nano-device the high level knowledge will be transmitted in a more aggregated (granulated) form. The knowledge includes besides the type of the activity being recognized on-line also the (fuzzy) rules based on which this recognition process has taken place as well as the recursive parameters based on which the on-line learning process can continue with new data.

In a real life experiment the training data is obtained from a single subject who is monitored in an automated manner. On the other hand, when several subjects are monitored, these experiments are carried out following some pre-defined instructions which make the recognition very accurate but they are based on restricted gesture models [3]. A real life experiment must allow a complete freedom of movement of the individual subjects of the experiment. In the experiments that are described in this paper both of these points of view are taken into account, because they have different domains of applications. In the former case, it could be useful for assisted live applications, for disabled people which perform similar gesture models. Alternatively, in the latter case, it is intended that a unique application is able to recognize a common pattern from different characters, which makes the definition

Both authors are with the Intelligent Systems Research Laboratory, Department of Communication Systems, InfoLab21, Lancaster University, Lancaster, LA1 4WA, UK; e-mail: j.andreu@lancaster.ac.uk

978-1-4244-8126-2/10/$26.00 ©2010 IEEE

of the pattern extremely complex. Moreover, we have to consider that our aim is to carry out this inference in *real-time*, which makes practically impossible to build a classification model *a priori*. Therefore, the approach that is taken is based on signal processing techniques which are adaptable and versatile so that recognition is based on **non-fixed** (***evolving***) models. In this study we develop and study the application of a recently introduced autonomous machine learning technique called evolving (fuzzy) rule-based systems [4,5,6] that provide the ability to take into account the dynamics and progressive evolution which the living subjects exhibit.

The paper is organized as follows. Section II describes our previous studies and other experimental works on this matter. Section III we explain the accomplished experiment and introduce the used techniques. Section IV explains with detail dataset collected and Section V pre-processing of data and feature extraction. Section VI elaborates on applied Evolving Fuzzy Systems (EFS). Section VII shows obtained results, VIII gives some details about alert recognition using same techniques and finally IX conclusions.

II. Background and Related work

In previous studies we worked on Ambient Assisted Living (AAL) application using wireless sensors network (WSN) for elderly monitoring [7]. Across our projects we employed a great variety of heterogeneous sensors such as RFID, bio-sensors, accelerometers, gyroscopes etc. We organized them in a cooperative framework [8]. The applications that were developed took a form of services. One of them was a fall detector that generates an alarm that is triggered when pre-defined thresholds are surpassed. By testing the fall detector prototype we realized that responses from the system and alarm generated were not effective when falls happened in an unusual manner, position or speed-up. As a result of this shortcoming we had to reconfigure related parameters on many occasions. This emphasizes the importance of algorithms and methods that are generic in nature and do not require user- or problem-specific thresholds, tuning but can instead adapt to the changing/evolving behavior of the subjects that are being monitored.

Our research was therefore directed towards study and development of generic in nature activity recognition methods and algorithms that are dynamically *evolving*, self-learning and *adapting* (as opposed to being pre-fixed and off-line).

There is a wide range of publications that report human activities recognition using wearable devices, e.g. using magnetic red switches [9], RFID Readers [10] and so on. However, the accelerometer attached to the body has been the predominantly used hardware. Some articles are focused on dealing with acceleration data such as [2] but they use several hoarders which were uncomfortable for monitored characters/subjects and even further away of a suitable prototype to be offered on the market.

In this paper we do compare the proposed approach and results of the study with the other published approaches which concern achieving an accurate activity recognition using just a single sensor. On this basis, an accurate activity classification technique using a single wrist-worn accelerometer has been developed [11]. In [12] a decision tree algorithm built from clusters created **beforehand** has been used which assumes an off-line experimental setting and pre-fixing the structure of the classifier which also implies that any future changes or evolution of the behavior by the subjects being monitored will not be correctly classified. Therefore, this algorithm is not suitable for *real-time* application.

III. The Proposed Approach

This section introduces the proposed approach for adaptive, *real-time* activity recognition from wearable wireless sensors. It comprise of several stages which are closely related. Success of recognition depends strongly on every single layer of the system. In what follows we will describe all stages that are included in the proposed self-learning classifier.

We used a wireless sensor with an accelerometer on board as a prototype. This sensor was a SunSPOT mote [13] which has a low-power, three-axial accelerometer that can be set to measure accelerations over a scale of ±6g. Moreover, the radio range is up to 10 meters (USA version allows up to 100m), but can be significantly less depending on other interference factors in the environment. Battery life is entirely dependent on the duty cycle (when running all the LEDs and the radio at full power will support less than a day (closer to 6 hours) of life; an almost continuous deep sleep can support well over a year).

Fig. 1. Smart wireless sensor attached to a subject's/character's arm

The core idea of the proposed paradigm is to infer higher level (human intelligible) knowledge (in the form of fuzzy rules) by processing on-line in a computationally efficient recursive manner in *real-time* streams of raw data locally on these wearable wireless sensors and only transmit the high level knowledge instead of transmitting the huge amounts of raw data directly. The proposed concept is centered at embedding local intelligence in wearable sensors borrowing some ideas from robotics. Emerging concept of *evolving* (intelligent) systems [4,5,6] has already been applied successfully to *intelligent* sensors in chemical industry [14], to robotic systems [15], etc. A hardware realization in a system-on-chip embedded device such as FPGA of the evolving clustering was also reported [16]. The proposed paradigm allows the personalization aspect to be addressed by extracting user-specific knowledge from the movement pattern extracted from the low level data streams (acceleration time series).

IV. Collecting Representative Samples

At the beginning of this research we needed to generate an accurate representative database for training the initial model. It was obtained from 18 subjects (10 men and 8 women). Supervised annotations were taken on during the experiment. All of experiments had duration between 60 and 90 minutes. The wireless wearable sensor was attached to the upper arm (figure 1) and maximum duration of monitoring was 20 minutes. Other studies claim that 20Hz of frequency is required to classify physical actives. In our case this frequency is slightly higher reaching 30 Hz of frequency.

A total of 18 activities were performed:

TABLE I LIST OF ACTIVITIES FROM THE EXPERIMENT

Activities list	
Walking	Carrying things while walking
Watching TV	Running
Washing	Tidying a room
Lying in a bed	Brushing teeth
Climbing stairs	Reading
Cycling	Vacuum cleaning
Standing	Exercising/stretching
Eating	Relaxing in a chair
Office work (in a chair)	Others

The conductor of the experiment was directing users to proceed with the selected activities as well as was taking note of the time when a new activity took place. Activities were performed in different order aiming a more realistic scenario and an increased complexity.

V. Processing Raw Data

Data collection is important, but processing the raw data is the key of a successful approach. First of all, the data are compressed. Raw data measured by two-dimensional accelerometer were used:

$$S_t = \{(x_t, y_t)\} \quad t = 1,2,... \tag{1}$$

Only the values when the derivative is equal to *0* (inflection point of the signal) are taken into account since they are the most significant points of the signal. This can be denoted as:

$$\begin{aligned} S_t = \{ & \{(x_k, y_k)\} \mid \\ & (y_{k-1} < y_k \wedge y_{k+1} < y_k) \vee (y_{k-1} > y_k \wedge y_{k+1} > y_k), \\ & y_{k-1}, y_k, y_{k+1} \in S_t \} \end{aligned} \tag{2}$$

In order to allow for a fair comparison of the proposed algorithm with other approaches we formed two types of experimental datasets:

1) Offline Dataset: This data set is based on the raw data from the sensors from each one of the characters/subjects. We gather a big dataset with all the data collected for each one of the characters and then we proceed with the feature extraction. In total, we extracted 72 features, $f_1,\dots,f_{72}$ (36 for each of the axes, *x* and *y*), namely:
   - variance;
   - kurtosis;
   - skewness;
   - 30 different percentiles;
   - correlation between axes data; correlation is defined as the ratio of the covariance and the product of the standard deviations for both axes, *x* and *y*:

   $$C = \frac{\mathrm{cov}(x, y)}{\sigma_x \sigma_x} \tag{3}$$

   - Energy; It is the sum of the squared discrete Fourier transform component magnitude of the signal divided by the length of the collected dataset:

   $$Energy = \frac{\sum_{i=1}^{w} |x_1|^2}{|w|}. \tag{4}$$

2) Online Dataset: In this case features were selected in a *real-time* fashion i.e. each time a new data was collected respective feature was updated recursively. In this sense, some approaches which are not recursive cannot be applied in on-line mode. In the case of Evolving Fuzzy Systems (EFS) [4,5] which is fully recursive approach this is the most appropriate mode. In total, we formed 66 different features ($f_1,\dots,f_{66}$); for each of the two acceleration axis's, *x* and *y*:
   - recursive mean: $\mu_t = \frac{t-1}{t}\mu_{t-1} + \frac{1}{t}x_t$; (5)
   - recursive variance:

   $$\sigma_t^2 = \frac{t-1}{t}\sigma_{t-1}^2 + \frac{1}{t-1}(x_t - \mu_t)^2; \tag{6}$$

   - *30* recursive percentiles:

   $$P_t^S = \mu_{t-1} + \frac{c^S \cdot \sqrt{\sigma_t^2}}{\sqrt{n}}; \tag{7}$$

   where *n* is the current number of samples, *S* is the required percentile. C^S is the value of the required percentile in a constant normal distribution.
   - current value of derivatives:

   $$\int_1^x \frac{1}{t} dx = \ln x. \tag{8}$$

VI. Evolving Self-Learning Fuzzy Rule-based Classifiers for Activity Recognition

A classifier is a mapping from the feature space to the class label space. The antecedent part of a FRB classifier describes the fuzzy partitioning of the feature space, $f \in R^{66}$ and with the consequent part - the class label, L_i, *i=[1, K]*. The structure of *eClass* [17] follows this typical construct of a fuzzy rule-based classifier:

$$\begin{aligned} R^i: \ & IF\left(f_1 \text{ is around } f_1^{i*}\right) AND \dots \left(x_{66} \text{ is around } x_{66}^{i*}\right) \\ & THEN \left(L^i\right) \end{aligned} \tag{9}$$

where $f = [f_1, f_2, \dots, f_{66}]^T$ is the vector of features; R^i denotes the i^{th} fuzzy rule; *i=[1, N];N* is the number of fuzzy rules; $\left(f_j \text{ is around } f_j^{i*}\right)$ denotes the j^{th} fuzzy set of the i^{th} fuzzy rule; *j=[1,n];* f^{i*} is the focal point of the i^{th} rule antecedent (note that this is a prototype - a real, existing data sample not the mean). L^i is the label of the class of the i^{th} prototype (focal point).

The inference in *eClass* is produced by 'winner takes all' rule:

$$L = L^{i*};\ i^* = \underset{i=1}{\overset{N}{\arg\max}}\,(\tau^i) \quad (10)$$

Where τ^i denotes the firing level of (degree of confidence in) the i^{th} fuzzy rule, which is determined as a product of the membership values, μ_j^i of the j^{th} feature, $j=[1,n]$ to the fuzzy set $\left(f_j \text{ is around } f_j^{i*}\right)$:

$$\tau^i = \prod_{j=1}^{n} \mu_j^i(x_j);\quad i = [1, N] \quad (11)$$

The membership functions that describe the degree of association with a specific prototype are usually of Gaussian form (characterized by good generalization capabilities and coverage of the whole feature space):

$$\mu_j^i = e^{-\frac{1}{2}\left(\frac{d_j^i}{r_j^i}\right)^2};\ i = [1, N];\ j = [1, n] \quad (12)$$

where d_j^i is th distance between a sample and the prototype (focal point) of the i^{th} fuzzy rule; r_j^i is the spread of the membership function, which also represents the radius of the zone of influence of the fuzzy rule.

A. Learning eClass

Typically, classifiers are trained off-line using evolutionary algorithms or gradient-based schemes such as back-propagation when combined with NN. The *eClass* family, however, is designed for on-line applications with an evolving (self-developing) FRB structure. To achieve this, the antecedents of the FRB are formed from the data stream around highly descriptive focal points (prototypes) in the input-output space, $z = \left[f^T, L\right]^T$ *per class*.

This on-line algorithm works similarly to adaptive control and estimation – in the period between two samples two phases are performed:

i) class prediction (classification);
ii) classifier update or evolution.

During the first phase the class label is not known and is being predicted; during the second phase, however, it is known and is used as supervisory information to update the classifier (including its ***structure evolution*** as well as its parameters update).

The main difference between *eClass* and conventional classifiers is;

i) the open structure of the rule-base (it self-develops on-line starting 'from scratch' while in a conventional classifier it is determined off-line and then fixed);
ii) the on-line learning mechanism which takes into account this flexible rule-base structure.

Note that the overall fuzzy rule base is composed of K sub-rule-bases so that in each sub-rule-base the consequents of all rules are the same, but the number of rules, N should be no less than the number of classes ($N \geq K$). That is, every new data sample with a class label that has not been previously seen becomes automatically a prototype. Since there is a prototype replacement and removal mechanism this is usually temporarily (this prototype is often later replaced by more descriptive prototypes). In this way, the classifier learns 'from scratch' and the number of classes does not need to be known in advance.

The basic notion of the partitioning algorithm is that of the *potential* or density in the feature space which is defined as a Cauchy function of the sum of distances between a certain data sample and ***all*** other data samples of class L in the feature space [18]:

$$D_k^l(Z_k) = \frac{1}{2 - \left[1/\sqrt{\sum_{j-1}^{n+1}(z_k^i)^2}\right]\sum_{j-1}^{n+1} z_k^i\, b_{k-1}^i} \quad (13)$$

$$k = 2,3,\ldots \qquad D_k^l(Z_1) = 1$$

Where $b_k^i = b_{k-1}^i + \sqrt{\frac{(z_k^i)^2}{\sum_{j=1}^{n+1}(z_k^j)^2}}$, $b_1^i = \sqrt{\frac{(z_1^i)^2}{\sum_{j=1}^{n+1}(z_1^j)^2}}$, $i = [1, n+1]$ and $z = [f^T, L]^T$ is the input/output vector.

Partitioning using the *density/potential* is based on the following principle: 'the point with the highest density is chosen to be the focal point of the antecedent of a fuzzy rule'. In this way fuzzy rules with high descriptive power and generalization capabilities are generated.

Each time a new data sample is read it affects the data density; therefore the potentials of the existing centers need to be updated. This update can also be done in a recursive way:

$$D_K^L(z_l^{i*}) = \frac{\left(N_k^l - 1\right) D_{k-1}^l\left(z_l^{i*}\right)}{N_k^l - 1 + \left[\left(N_k^l - 2\right)\left(\frac{1}{D_{k-1}^l\left(z_l^{i*}\right)} - 1\right) + d\left(z_l^{i*}, z_k\right)\right]} \quad (14)$$

Once the density/potential of the new-coming data sample is calculated *recursively* using (11) and the density/potential of each of the previously existing prototypes is recursively updated using (14) they are compared. If the new data sample has a *higher* potential than any of the previously existing prototypes of that class, L then it is a good candidate to become a focal point of a new rule in this sub-rule-base because it has high descriptive power and generalization potential:

$$D_k^L(z_k) > D_k^L\left(z_L^{i*}\right),\ \forall i^* \in N^L \quad (15)$$

Forming a new fuzzy rule around the newly added prototype leads to a *gradual* increase of the size of the sub-rule-base, which is why the approach is called '***evolving***':

$$z^{\left(N^L+1\right)} \leftarrow z_k \quad (16)$$

The potential of the newly generated rule is set to *1* temporarily (it will be updated to take into account the influence of each new data sample on the generalization potential of this particular focal point by (15) with each new data sample being read):

$$D_k^L\left(z^{\left(N^L+1\right)*}\right) \leftarrow 1 \quad (17)$$

To increase the interpretability and update the rule base one needs to remove previously existing rules that become ambiguous after insertion of a new rule. Therefore, each time a new fuzzy rule is added it is also checked whether any of the already existing prototypes in this sub-rule-base are described by this rule to a degree higher than e^{-1}:

$$\exists i, i = [1, N^L]; \quad \mu_i^j(z^{N^L+1}) > e^{-1} \quad \forall j, j = [1, n] \tag{18}$$

If any of the previously existing prototypes satisfy this condition the rules that correspond to them are being removed from this sub-rule base (in fact, replaced by the newly formed rule).

The spreads of the membership functions of the sub-rule-base of the respective class are also recursively updated based on the data distribution:

$$r_k^i = \rho r_{k-1}^i + (1-\rho)\sigma_{k-1}^i; \; i = [1, N^L] \tag{19}$$

where ρ is a learning parameter; it determines how quickly the spread of the membership functions will converge to the local scatter of that cluster; the default value is *0.5*

VII. Experimental Results for Activity Recognition

In this section the experimental results using the proposed approach will be described and compared with other existing published studies and also algorithms used in other investigations in the activity recognition field.

We separated the experimental part in two tests. First, in an *off-line* mode we tried to classify from the *off-line* dataset which activity happened in a certain moment of the past. On the other hand, we used the *on-line* dataset from just one character to recognize the activity (s)he is performing/ engaging in a *real time* fashion.

A. *Offline Test*

A dataset from 18 subjects was collected. A standard *10 fold cross validation* test was carried out in order to assess the performance of a range of widely used off-line classification techniques. This test is used in order to evaluate how a particular approach will ***generalize*** to a validation data set [18].

- *BayesNet:* Bayesian Network classifiers [19] encode probabilistic relationship among variables of interest. An estimator which estimates conditional probabilities directly from data was used once the structure of the Bayesian network has been learned off line;
- *ADABoostM1:* AdaBoost.M1 accesses to a learning algorithm (which the developers generically title WeakLearn) that calls repeatedly different distributions of the training data set. WeakLearn calculates a hypothesis, or classifier, that attempts to correctly classify all instances of the test data [20];
- *J48:* It is a version of an earlier algorithm developed by J. R. Quinlan, the very popular C4.5 [21].
- *J48 Rotation Forest:* is a method for generating classifier ensembles based on feature extraction [22]. To create the training data for a base classifier, the feature set is randomly split into *K* subsets (*K* is a parameter of the algorithm) and Principal Component Analysis (PCA) is applied to each subset. All principal components are retained in order to preserve the variability information in the data. Thus, *K* axes rotations take place to form the new features for the base classifier. Rotation Forest can do classification and regression depending on the base learner. In our case, the base selected classifier is a C4.5 tree;
- *Support Vector Classifier:* We tested the John Platt's sequential minimal optimization algorithm for training a support vector classifier. It solves multi-dimensional class problems using pair-wise classification [23]. We used regression models like outputs of the support vector machine (similar to the eClass1 first order local linear models [17]). In the multi-class case the predicted probabilities are coupled using Hastie and Tibshirani's pair-wise coupling method [18];
- *Neural Networks:* A classifier that uses the popular error back propagation learning technique was designed using a multilayer perceptron framework. It uses three or more layers of neurons (nodes) with nonlinear activation functions;
- *KNN Classifier:* It is an instance-based classifier which operates on the premise that classification can be done by relating new samples with the older ones, according to some distance or similarity function [19].

In a real life and real time applications, the classification rate is not the only factor that is important. The amount of time the classifier requires to be developed or trained is also of a paramount importance. Some of the classifiers, for example, the Neural Network classifier and Rotation Forest take more than 1 minute to build the model/classifier. This constraint makes these algorithms inefficient for *real time* purposes. Another vitally important aspect of an algorithm is the memory requirements. Most of them require significant memory storage (kNN and EFS are the only exceptions). The iterative nature is also a big restriction that is typical for some of the algorithms (most notably ADABoost, NN and SVM) which also hampers real-time on-line applications.

Comparison results with all the selected approaches are displayed in Figure 2.

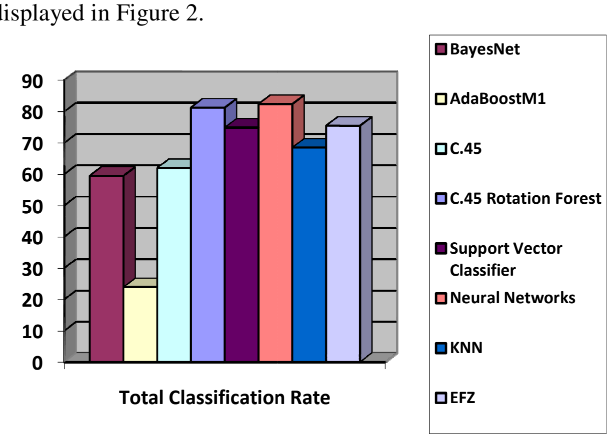


Fig. 2 Comparative results with different algorithms.

Although EFS is on-line by virtue of its nature, we compared it using off-line mode (assuming we stopped its learning and used classification/predictions only) for the sake of a comparison with other off-line methods. It has to be noted that EFS demonstrated very good results that were pretty close to what other robust off-line classifiers like Neural Network and Rotation Forest get and even slightly superior to Support Vector Machine classifiers. This is a very good result having in mind that EFS is one pass, recursive, computationally light and designed primarily for low constraint devices and real time applications.

B. Online Test

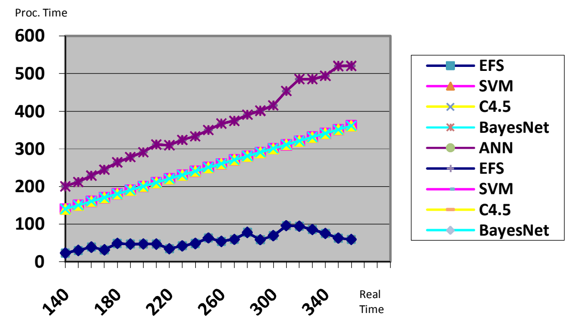
On the off-line test described above we aimed to mainly put the proposed technique in the context of other existing (primarily off-line) techniques. By having the results that the proposed technique is comparable and superior quality than other off-line techniques in terms of precision, we concentrated ourselves on the advantages of the on-line application such as computational simplicity (low memory requirements, short time of response, re-learning ability). In this test we will also employ the on-line feature extraction capability of EFS.

TABLE II
ONLINE DATASET

Training			
Subjects	Activities	Total Time	Data Samples
1	3	≈6 min	≈91,510
Testing			
Subjects	Activities	Total Time	Data Samples
1	1	≈20 sec.	≈5,484

It should be noted that an on-line experiment has some specifics that are generic for all approaches that are applied. On-line approaches needs to create some stable patterns (fuzzy rules in the case of EFS) for classification which might require a relatively long period of time if these activities have similar motion patterns. On this basis, to include '*Watching TV*' and '*Sitting in a Chair*', followed in the same action sequence, would result in the classifier not being able to distinguish between both to a sufficient degree. In order to get significant results in an effective fashion we carried out an experiment with a total length of 6 minutes. In this experiment, 3 unrelated activities were performed in the following order: '*Climbing Stairs*', '*Exercising/Stretching*' and '*Walking*'. Between the former and the latter there is a similar motion pattern. For this reason, we included the activity '*Exercising/Stretching*' between the two. All experiments were performed ten times and the results were averaged. In the following experiment the dataset was composed of data from a single character/subject.

According to real-time and sensors requirements, response time should be the minimum possible in order to flush data buffers as soon as possible. The more the time of operation is constant the more suitable is the approach for a real-time operation. This is because we should define operational time constraints that allow us to set up a maximum time of response.



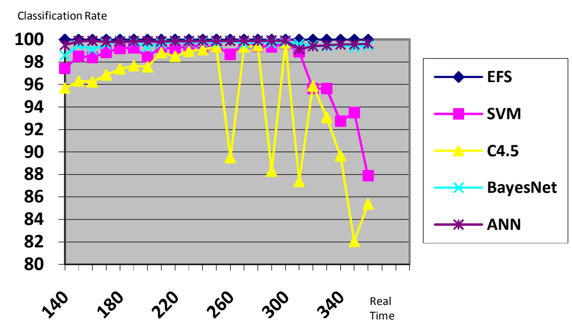


Fig. 3 a) top plot - time needed for calculations (vertical) versus the actual(real) time of the experiment (both in seconds); b) bottom plot – classification rate in percentage (vertical) versus actual (real) time of the experiment (in seconds).

Figure 3 a) (top plot) shows the time required for the respective algorithms to calculate the results in seconds (the vertical line) versus the actual (real) time of the experiment for the 140th to 360th seconds of the experiments (we registered the data stating form the 140th second because at this moment a new activity started). We consider that it is not interesting to carry out the comparison when we were only getting data from a single activity. Therefore, periods of time displayed correspond to a transition between different activities as follows:

- [140,310], s: *Climbing Stairs* → *Exercising/Stretching*
- [310, 360], s: *Exercising/Stretching* → *Walking*

From Figure 3 b), bottom plot one can see that it takes to build an ANN classifier 40% more time than the real time of the experiment which makes it prohibitive for a *real-time* application. Responses of a number of such algorithms come delayed and response time increases exponentially. SVM, C4.5 and BayesNet increase the time of their response continuously. In addition, ANN needs to gather and keep all previous data in order to build a classification model in any particular time instant. As opposed to that, EFS simply uses the last data sample to update the model rather than to re-build it each time. The response time of EFS oscillates between 23s and 93s, this rate keeps uniformity though. It increases when a new rule/activity has been identified but immediately afterwards, for new data samples, time of response becomes lower, until a new rule is formed again.

All algorithms provide a very good classification rate on the fist activity. However, when the second activity takes place (around 310th s) some of the algorithms (most notably C4.5 and SVM reduced their classification rate significantly, see Fig. 3 b). BayesNet keeps a similar classification rate as EFS. However, in *on-line* mode EFS is fairly superior maintaining a classification rate very close to 100%. It is on condition that this classification rate becomes slower across the time and new activities are detected. However, evolving techniques help the system to recover its accuracy, as a result of its adaptive behavior.

VIII. ALERT RECOGNITION

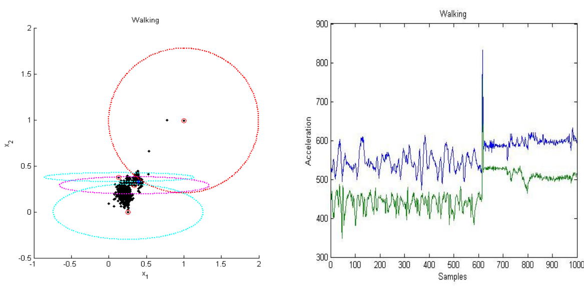


Fig 4. a) left plot - Raw data for the activity *Walking*, and b) right plot - Resultant clusters from processed raw data with EFS.

During our study we reckoned that there were various outliers identified. They do not form a significant part of the pattern of the activity, but they are important indicators of possible sensor faults or falls of the character/subject. Thus, we observed the time when they appeared and then we discovered that it is related to some special situations such as sensor failure, disconnection or whatever fatality. Accordingly, an important hypothesis was studied aiming to model such cases by new rules in order to get good event recognition of any non-expected situation that takes place in the observed character/ subject. This event recognition can be very useful in healthcare and ambient assisted living applications as well as in defense and sport.

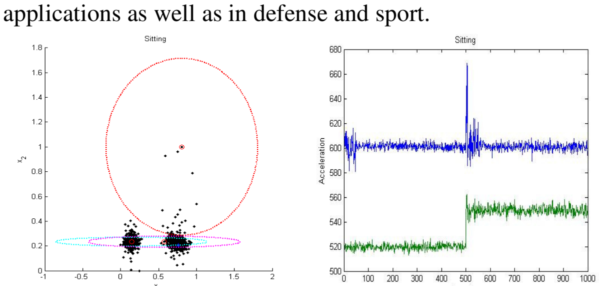


Fig 5. a) left plot - Raw data for the activity '*Sitting*'; b) right plot - Resultant clusters from processed raw data with EFS.

For the following experiment individuals were asked to fall down in a mat after a pre-established circuit. Tree activities were selected to be performed during the supervised circuit. The individual performs 3 falls as long as he was executing the required activity.

In the following graphs (Figures 4-6) we can see the evolution of raw data when a fall is performed. Selected activities were '*Walking*' (Figure 4), '*Sitting*' (Figure 5) and '*Climbing Stairs*' (Figure 6) respectively.

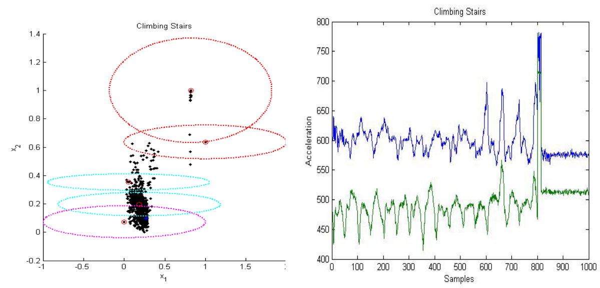


Fig 6. a) Raw data for the activity '*Climbing stairs*'; b) right plot - Resultant clusters from processed raw data with EFS.

All of these graphs have different waveforms as a result of different motion actions performed. The event is easily visible in the graphs as a peak (maximum) of the function. Nevertheless, in a *real time* application we have to know that those data samples denote a maximum fluctuation in the wave, which means anomalous event during the course of the experiment. However to achieve this aim, we have to consider some meaningful indicators during the course of the monitoring so that we can figure out rules for this maximum according to an impact value per data sample. It is called potential (as mentioned above) which represents the data density and is the basis of the formation of new rules in the model/classifier.

Motion events may appear in completely different forms depending on the activity and also depending on the monitored subject. It is important to emphasize those factors such as age, gender, life style and so on, influence the appearance of such events. The reason of it is that is difficult or impossible to take into account all such factors; hence the use of adaptive and evolving techniques such as EFS is critical for a well designed application. The use of techniques based on fixed threshold can be a weak approach which cannot be extended to real activity conditions because it will be biased towards a specific person or persons and never be generic (it will be subject- and problem-specific).

Acceleration waveform usually changes when a fall has taken place, because the physical posture of the monitored character obviously changes as well as the normal and gravitational acceleration.

In Figures 4-6 b) (right plots) there are clusters plots regarding raw data shown in Figures 4-6 a) (left plots). The colours of each one of the clusters represent different motion patterns (the blue colour corresponds to the starting activity; red color corresponds to the data samples that are directly related with the anomalous event; and, finally, violet color represents the new activity which comes after the event has been performed).

In the case of '*Walking*' most of data can be found between two well delimited cluster centers (see Figure 4b). Nevertheless, when a fall takes place, data start to come with a slight increase of acceleration. This new data establishes a new rule in the model/classifier that is interpreted as new physical posture. It is important to note that a new physical posture comes after an event. If the change of posture has not been soft enough and points with considerable potential came previously, it might mean a significant event inside the model.

On the other hand, during '*Climbing stairs*' the fall that was performed by the character/subject reached a greater level of complexity. Therefore, in this case two clusters represent the event (see Figure 6b).

In order to reflect the appearance of a new event new rules are formed around new prototypes as described above. They help the overall classifier to be tuned to the new data pattern.

If an outlier data point has been discovered, it could be an anomalous behavior. Therefore, we check the local density/potential to identify the local significance of this change of the motion data pattern and then to be able to identify a new event or even to classify the data as an outlier (possibly due to sensor fault. When the subject is moving the window of time move through, the local density/potential calculated can be larger than 1s. In general, it varies between 1 and 3s. In this time a new rule is formed as a result of the new acceleration scheme induced by the postural change. Having into account the coming out of an outer data point and a rule in the next second, this is enough information to consider that an event such a fall has taken place.

IX. CONCLUSION

To sum it up, real-time evolving classifier (eClass) has been used in this research to recognize activities from two-axial accelerometer data, collected by smart sensor nodes. Results were very successful, getting almost a 99% by delineating 3 unrelated activities. For example regarding *Running, Walking, Running, Sitting* or *Climbing Stairs*, acceleration patterns in connection with gravity were easily identifiable after a short period of pre-training. However, there are complex activities such as *Cycling, Tiding, Cleaning*, *Exercising* etc. which are comprised of several another basic activities that are not easy to recognize in real-time after a short pre-training time. We also aim at classifying these complex activities, so next research steps would be to study in depth the transitions between different activities in order to work out delimited states that determine those complex activities. Another direction of future research is linked to collaborative classifiers in a network of diversified smart sensors.

REFERENCES

[1] M. Kim "ADL Monitoring System Using FSR Arrays and Optional 3-Axix Accelerometer" in *Proc. 5597 Ambient Assistive Health and Wellness Management in the Heart of the City.* pp. 217-224, 2009.

[2] L. Bao and S.S. Intille, "Activity Recognition from user-annoted acceleration data" *Pervasive*, pp.1-17, *2004*.

[3] J. Mantyjarvi, J. Himberg and T. Seppanen, "Recognizing human motion with multiple acceleration sensors" *IEEE Intern. Conf. on Systems*, *Man, and Cybernetics*, pp. 747-752, 2001.

[4] P. Angelov, "Evolving Rule-based Models: A Tool for Intelligent Adaptation", *Proc. 9th IFSA World Congress*, Vancouver, BC, Canada, 25-28, pp.1062-1067, July 2001.

[5] P. Angelov, "An Approach for Fuzzy Rule-base Adaptation using On-line Clustering", *International Journal of Approximate Reasoning*, vol. 35 , No 3, pp. 275-289, March 2004.

[6] P. Angelov and D. Filev, "An Approach to On-line Identification of Takagi-Sugeno Fuzzy Models", *IEEE Trans. on System, Man, and Cybernetics, part B - Cybernetics*, vol.34, No. 1, pp. 484-498, 2004.

[7] J. Andreu, J. Viúdez and J.A. Holgado, "An Ambient Assisted-Living Architecture based on Wireless Sensor Networks." *Advances in Soft Computing,* Vol 51, pp. 234-248, 2008.

[8] J. Andreu, J.A. Álvarez, A. Fernández-Montes and J. A. Ortega, "Service-Oriented Device Integration for Ubiquitous Ambient Assisted Living Environments", *International Work-Conference on Artificial Neural Networks IWANN.* No. 10, pp. 843-850, 2009.

[9] B. Logan, J. Healey, M. Philipose, E. Munguia Tapia and S.S. Intille: "A Long-Term Evaluation of Sensing Modalities for Activity Recognition". *Ubicomp,* pp. 483-500, 2007.

[10] M. Buettner and R. Prasad "Recognizing daily activities with RFID-based sensors". *Proc.of the 11th international conference on Ubiquitous computing,* pp. 51-60, 2009.

[11] E. Haapalainen, P. Laurinen and P. Siirtola, "Exercise energy expenditure estimation based on acceleration data using the linear mixed model". *Information Reuse and Integration*, pp. 131-136, 2008.

[12] P Siirtola, "Clustering-based activity classification with a wrist-worn accelerometer using basic features". *Proc. IEEE Symposium on Computational Intelligence and Data Mining.* pp. 95-100, 2005.

[13] Sun Microsystem's SunSPOT. Available: http://www.sunspotworld.com

[14] P. Angelov and A. Kordon, "Adaptive Inferential Sensors based on Evolving Fuzzy Models: An Industrial Case Study". *IEEE Trans. on Systems, Man and Cybernetics – part B, Cybernetics,* 2010.

[15] P. Angelov and X. Zhou, "Evolving Fuzzy Classifier for Real-time Novelty Detection and Landmark Recognition by a Mobile Robot". *Studies in Computational Intelligence Series*, pp. 95-124, 2007.

[16] M. Everett, P. Angelov, *EvoMap: On-Chip Implementation of Intelligent Information Modelling using EVOlving MAPping,* Lancaster, UK: Lancaster University, 2005.

[17] P. Angelov and X. Zhou, "Evolving Fuzzy-Rule-based Classifiers from Data Streams", *IEEE Trans. on Fuzzy Systems*, vol. 16, No. 6, pp.1462-1475, Dec. 2008.

[18] T. Hastie and R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference and Prediction.* Ed. Heidelberg, Germany: Springer Verlag, 2001.

[19] J. Pearl and S. Russell, "Bayesian Networks". *Handbook of Brain Theory and Neural Networks*. pp. 157–160. USA: MIT Press, 2001.

[20] R.E. Schapire. "The boosting approach to machine learning: an overview. Nonlinear Estimation and Classification", *Lecture Notes in Statist.,* Vol. 171, pp 149-171. 2003.

[21] J. Ross, C4.5: Programs for Machine Learning, *Morgan Kaufmann,* 1993.

[22] J. Rodriguez, "Rotation Forest: A New Classifier Ensemble Method". *IEEE Transactions on Pattern Analysis & Machine Intelligence* Vol. 28, pp. 1619-1630, 2006.

[23] T. Hastie, S. Rosset, R. Tibshirani, J. Zhu, "The Entire Regularization Path for the Support Vector Machine", *Journal of Machine Learning Research* , pp.1391-1414, 2004.